коэффициенты повышения стойкости системы резервного копирования данных и системы защищенной электронной почты к компрометации ключей при использовании биометрии.

Полученные данные позволяют сделать утверждение о повышении криптографической стойкости систем к компрометации за счет введения в эти системы криптографических ключей, с внедренной в них биометрической информацией участников.

СПИСОК ЛИТЕРАТУРЫ:

1. *Севастьянов Б. А.* Курс теории вероятностей и математической статистики. М., 1982. — 256 с.
2. *Половко А. М., Гуров С. В.* Основы теории надёжности. СПб., 2006. —704 с.

Е. А. Лавринов
Московский инженерно-физический институт (государственный университет)

РАЗРАБОТКА МОДЕЛИ И АЛГОРИТМОВ СИСТЕМЫ ДИСТАНЦИОННОГО ВЫПОЛНЕНИЯ КРИПТОГРАФИЧЕСКИХ ОПЕРАЦИЙ

Представлены эффективные протоколы дистанционного шифрования, выполнения проверки целостности данных, генерации и проверки подписи. Разработана модель СКЗИ для дистанционного выполнения криптографических операций.

В настоящее время наиболее уязвимым местом в шифровании является не генерация ключей или использование функций генерации ключей, а управление ключами, используемыми в процессе шифрования. Одни и те же ключи используются для зашифрования и расшифрования данных, поэтому их нужно надежно охранять, чтобы защитить данные. В то же время прикладные программы и пользователи должны иметь доступ к ключам, чтобы расшифровать данные для нормального использования. Эта проблема решается путем физического разделения устройства хранения ключей и устройства, выполняющего криптографические операции. Появляется потребность в криптографических протоколах для дистанционного выполнения операций.

Решением данной проблемы является разработка системы дистанционного выполнения криптографических операций, которая включает в себя следующие протоколы:

1. Протокол дистанционного шифрования данных.
2. Протокол дистанционного выполнения проверки целостности данных.
3. Протокол генерации и проверки ЭЦП.

В качестве устройства хранения ключа удобнее использовать интеллектуальную карту (ИК), которая является важным строительным блоком во многих современных прикладных программах (ПП), обеспечивающих безопасность. В частности, взломозащищенный модуль, низкая стоимость, присущая портативность и бесконтактное соединение с терминалом делают ИК особенно привлекательной для использования ее в качестве хранилища секретных ключей, если терминал является незащищенным. С другой стороны, эти же свойства ограничивают использование ИК. Бесконтактное соединение с

терминалом и низкая стоимость ИК означают, что карта не может обрабатывать данные с такой же производительностью, как и терминал.

Для некоторых ПП, в которых применяются протоколы аутентификации пользователя путём проверки правильности его реакции на непредсказуемый запрос системы, и цифровые подписи хэш-кодов сообщений, низкая производительность ИК неприемлема; секретный ключ, который хранится на ИК, используется достаточно редко и требования по скорости минимальны. Однако в других ПП, включающих файловое шифрование, зашифрованный трафик, мультимедиа и видео, с помощью секретного ключа ИК зашифровывается и расшифровывается гораздо больший объем трафика. В данном случае производительности ИК может не хватить, так как она зависит от скорости системы, ограниченной задержками и производительностью интерфейса карты, а также вычислительной возможностью карты.

Поэтому необходимо переносить часть работы с медленной и вычислительно ограниченной ИК на более быстрый и мощный терминал. Приходится идти на компромисс между безопасностью, быстродействием и ценой. В одном случае можно создать ИК и интерфейс таким образом, чтобы ее быстродействие соответствовало быстродействию процессора, расположенного на терминале. Но это не всегда технологически выполнимо и может заметно увеличить стоимость всей системы. В другом случае можно использовать карту только для хранения ключа, а выполнять все криптографические операции на терминале за счет копирования ключа в процессор терминала. Предоставление ключа влечет за собой изменение в модели безопасности ИК, так как процессор терминала должен быть надежным, чтобы защитить ключ.

Для ПП, в которых требуется высокая производительность для шифрования большого объема данных с использованием ИК, основанных на управлении ключами, необходимо использовать схему, которая переносит всю работу на процессор терминала, причем, не увеличивая требования доверия к терминалу. Работы M. J. Beller, L. Chang и Y. Yacobi [1] в области асимметричной криптографии сосредоточены на криптосистемах с открытым ключом, в которых части, требующие вычислений, могут быть перенесены из одной общей части в другую независимую часть, но это не решает данную задачу. Другая работа D. Beaver и J. Feigenbaum [2] касается скрытых объектов конкретных типов распределенных вычислений и не может быть использована для шифрования с помощью блочных шифров. В работе Matt Blaze [3] предложен простой протокол RKEP (Remotely Keyed Encryption Protocol), в котором используется криптосистема с секретным ключом, где медленно, но безопасно происходит перенос большей части работы из ИК в незащищенный, но быстрый процессор терминала. Данный протокол позволяет значительно увеличить скорость шифрования информации по сравнению с обычным методом, когда вся шифруемая информация проходит через ИК.

Большинство работ в направлении проверки целостности данных, хранящихся в памяти, основаны на использовании дерева Merkle [4], которое изначально было предложено как способ подтверждения подлинности данных между объектами, не являющимися доверенными, с использованием минимального объема памяти. Blum и др. [5] впервые предложили использовать хэш-деревья для проверки подлинности данных, хранимых в не являющейся доверенной памяти большого объема, а также представили теоретические доказательства безопасности. Впоследствии основанный на использовании хэш-дерева метод проверки целостности данных памяти был использован несколькими способами:

- для построения надежных баз данных [6], использующих незащищенную память;
- для управления постоянным состоянием в цифровых системах управления правами [7];
- для проверки структур данных, таких как стеки и очереди, хранимых в не являющейся доверенной памяти в системах с ограниченной памятью, таких как ИК [8];
- для подтверждения выполнения программ доверенным процессором [9].

Хотя объем защищенной памяти при использовании схем, основанных на хэш-дереве, мал ($O(1)$), так как только хэш-код корня хранится в защищенной памяти), число запросов к незащищенной памяти для проверки целостности данных составляет $O(\log_m N)$, где m — число дочерних узлов для каждого

узла, а N — число листьев хэш-дерева (а также число блоков в памяти, если на один лист приходится один блок). Широкое использование хэш-деревьев вызвало исследование, нацеленное на смягчение недостатков работы схем проверки целостности данных памяти, основанных на использовании хэш-деревьев. Gassend и др. [10] предложили архитектурные изменения кэш-памяти с целью сокращения временных затрат методов проверки целостности, основанных на использовании деревьев. Williams и Sirer [11] используют аналитическое моделирование, чтобы определить размер листа дерева, то есть, число блоков памяти в листе, которое приведет к оптимальной работе. Обе работы [10, 11] улучшают производительность с помощью воздействия на некие параметры, влияющие на работу, и не предлагают никаких концептуальных изменений.

Метод проверки целостности данных с использованием запросов к незащищенной памяти [12] уменьшает количество запросов к незащищенной памяти по сравнению с существующими методами, предлагая концептуальное продвижение, основанное на численно-теоретическом подходе. Кроме того, емкостные затраты для реализации уменьшения количества запросов к незащищенной памяти достаточно малы.

Атаки на систему возникают, как правило, из-за раскрытия секретных ключей, что приводит к полной потере безопасности. Эта проблема представляет собой большую угрозу: на практике для злоумышленника легче получить секретные ключи от наивного сотрудника, чем использовать вычислительные возможности для взлома системы. В настоящее время угроза увеличивается из-за того, что люди используют мобильные устройства, которые имеют удаленный доступ из открытых или чужих доменов.

Для решения этой проблемы существуют два класса методов [13]. В первом случае попытка полностью предотвратить разглашение ключа. Хотя это и является важной задачей, но это не всегда практично. Например, когда используют портативные устройства для выполнения криптографических операций (например, шифрование с использованием мобильного телефона), можно ожидать, что устройство может быть само физически скомпрометировано каким-нибудь способом (потеряно или украдено), в этом случае разглашение ключа неизбежно. Более того, чтобы полностью избежать разглашения ключа — даже если это не мобильные устройства — обычно требуется определенная степень физической защиты, которая может быть как дорогой, так и неудобной. Второй подход предполагает, что разглашение ключа когда-нибудь произойдет и пытается минимизировать ущерб, в случае попадания ключей к злоумышленнику. К данному классу относятся: разделение секрета, пороговая криптография, действенная криптография, криптография, связанная с защитой от разглашения ключа, и подпись с опережающей безопасностью.

Из выше рассмотренного для удаленного зашифрования и расшифрования данных будет использоваться протокол RKEP, поскольку только он позволяет значительно увеличить скорость шифрования информации по сравнению с обычными методами, когда вся шифруемая информация проходит через ИК [14]. С точки зрения осуществления проверки целостности данных определенно приоритет имеет методика проверки целостности данных с использованием запросов к незащищенной памяти [12], так как она значительно уменьшает коммуникационную сложность проверки, не вызывая соответствующего увеличения емкостной сложности.

Для протокола RKEP [3] участниками являются терминал и ИК. Терминалу необходимо зашифровать и расшифровать большие блоки секретным ключом, который хранится на ИК. Несмотря на то, что терминалу по определению доверено обрабатывать открытый текст, которым он владеет, ему не известен секретный ключ. ИК известен ключ K, но карта ограничена вычислительно и по производительности и не может обрабатывать целые блоки за то время, которое необходимо терминалу. Протокол RKEP позволяет терминалу выполнять одно взаимодействие с ИК, не требующее высокой производительности фиксированного размера, чтобы терминалу получить достаточно информации для зашифрования и расшифрования блока произвольной длины. Тем не менее, терминал не может зашифровать и расшифровать другие блоки без оперативного доступа к ИК, даже имея предыдущий доступ к карте.

RKEP требует, чтобы терминал и ИК разделяли алгоритм блочного шифра, например, DES, который оперирует с n-битовыми шифрблоками и шифруется k-битовым ключом, а, именно, нет требования, где терминал и ИК выполняют одну и ту же шифр-функцию; тем не менее, если используются два шифра, безопасность системы ограничена слабейшим шифром. Между терминалом и картой должен быть безопасный (секретный) канал. Терминал оперирует с большими блоками открытого текста (P) и шифртекста (C).

Работа *протокола RKEP* состоит в следующем. Терминал в цикле вычисляет из блоков открытого текста P_i блоки «промежуточного» шифртекста I_i, после чего посылает I_1 ИК. Она вычисляет для него шифртекст C_1 и ключ шифрования K_P, затем передает их терминалу. Терминал в цикле вычисляет все остальные блоки шифртекста C_2, ... , C_n на ключе K_P в режиме сцепления блоков. Расшифрование выполняется похожим образом: терминал пересылает карте блок шифртекста C_1, ИК вычисляет для него «промежуточный» блок открытого текста I_1 и ключ рашифрования K_P, терминал, получив это сообщение, расшифровывает все остальные блоки шифртекста.

Проверка целостности данных памяти относится к процессу определения любого НСД к данным, хранящимся во внешней памяти. Эта проверка является важной частью безопасной и надежной архитектуры обработки данных. Большинство предлагаемых архитектур для безопасной и надежной обработки данных включают защищенный от постороннего вмешательства процессор с кэш-памятью для хранения криптографических ключей, высокочувствительного кода и данных (например, безопасное ядро), а также специализированное криптографическое аппаратное обеспечение для проведения эффективных криптографических вычислений. Однако код и данные доверенных ПП, которые не могут поместиться в ограниченной, но защищенной кэш-памяти, передаются на внешнюю память, которая расположена вне периметра, защищаемого взломоустойчивой оболочкой процессора. Таким образом, необходима схема проверки целостности данных памяти, которая может обнаружить НСД к данным в период времени между записью данных во внешнюю память и их чтением из неё защищенным процессором.

Предполагается, что внешняя память, содержащая программный код и данные, находится в полном распоряжении противника, который может изменять значения в любом участке памяти различными способами. Проверка целостности данных памяти совершается программой, запущенной на защищенном процессоре, и обеспечивает целостность кода и данных, запрашиваемых любой программой, работающей в нормальном режиме. Для того чтобы проверить целостность значений, расположенных на внешней памяти и запрашиваемых программами, программа проверки делает запросы к внешней памяти для получения дополнительных данных. Количество таких запросов, сделанных программой проверки, определяется как ее (проверочной программы) *коммуникационная сложность*. Программа проверки имеет собственную выделенную кэш-память, используемую для проверки целостности данных на незащищенной внешней памяти. Размер кэш-памяти, используемой программой проверки, называется *емкостной сложностью*. Два основных требования к любой эффективной программе проверки целостности данных памяти заключаются в следующем:

· программа должна обнаруживать любое несанкционированное изменение значений, хранимых во внешней памяти;

· важно, чтобы проверка целостности кода и данных, запрашиваемых исполняемой программой, не занимала много времени.

Для того чтобы программа проверки целостности работала эффективно, коммуникационная сложность должна быть как можно меньше, поскольку множественный доступ к памяти требует больших затрат времени. Однако уменьшение коммуникационной сложности приводит к увеличению емкостной сложности. Этот факт можно формально описать следующим образом. Пусть внешняя память состоит из n блоков, которые объединены в наборы по m блоков, криптографическое хэширование применено ко всем $\left\lceil \frac{n}{m} \right\rceil$ наборам, и результатом являются $\left\lceil \frac{n}{m} \right\rceil$ хэш-кодов, которые хранятся в кэш-памяти

программы проверки. Пусть блок памяти, принадлежащий i-му набору, считывается рабочей программой, тогда для проверки целостности прочитанного блока, должны быть считаны оставшиеся (m-1) блоков в i-м наборе, чтобы можно было вычислить хэш-код i-го набора. Вычисленный хэш-код сравнивается с истинным хэш-кодом i-го набора, который безопасно хранится на процессоре. Это совпадение означает, что набор (следовательно, и блок) не был подделан. Емкостная сложность в этой схеме равна $O\left(\frac{n}{m}\right)$, а коммуникационная сложность составляет $O(m)$. Они обратно пропорциональны друг другу. Оригинальным решением с емкостной сложностью $O(1)$ является дерево Merkle [4] для ПП, которые требуют минимальных емкостных затрат. Коммуникационная сложность схемы на основе дерева Merkle $O\left(m + \log_2\left(\frac{n}{m}\right)\right)$ может быть достаточно значима для больших значений n. Более того, уменьшение m снижает одно слагаемое коммуникационной сложности дерева Merkle, но увеличивает другое. Таким образом, существует такое уменьшение m, после чего это становится неэффективным. Однако в силу существующих тенденций к увеличению разрыва между скоростями работы процессора и памяти, высокая вычислительная сложность схемы проверки целостности памяти может значительно увеличить задержку между запросами к памяти. Благодаря увеличению процессорной памяти можно допустить различные величины емкостной сложности. Однако очень желательно, чтобы коммуникационная сложность проверки целостности памяти была низкой.

Для уменьшения коммуникационной сложности в рассматриваемой методике используется Китайская теорема об остатках [15, 16].

Существенная особенность предлагаемой схемы состоит в том, что коммуникационная сложность является независимой от емкости памяти n и размера набора m и зависит только от константы k, которая в свою очередь не зависит от n и m. При таком сокращении коммуникационной сложности емкостная сложность схемы составляет .

При определении схемы ЭЦП, работа которой будет осуществляться в системе дистанционного выполнения криптографических операций, можно выделить три модели: схема с опережающей безопасностью, схема с изоляцией ключа [13] и схема, стойкая к вторжениям [17], которые не относятся к обычным схемам ЭЦП, где цифровые подписи имеют фундаментальное ограничение: если секретный ключ подписывающего скомпрометирован, то все документы, подписанные им в прошлом или те, которые еще будут подписаны, становятся сомнительными.

Подход, используемый при создании опережающей безопасности, изменяет периодичность секретного ключа и требует от подписывающего должным образом удалять старые секретные ключи. Таким образом, время использования открытого ключа разбивается на периоды, в конце каждого периода генерируется новый секретный ключ, а старый надежно удаляется. Номер периода времени, когда документ был подписан, входит в состав подписи и является входным параметром для алгоритма проверки подписи. Подписи с некорректным периодом времени являются неверными.

Цель схем ЭЦП с опережающей безопасностью — сделать выгодным частую смену секретного ключа без соответствующей замены открытого ключа.

Схема с изоляцией ключа заключается в следующем [13]: пользователь начинает с регистрации единственного открытого ключа PK. Секретный мастер-ключ SK^* хранится на устройстве, которое физически защищено и является стойким к компрометации. Все операции расшифрования, тем не менее, выполняются на незащищенном устройстве, для которого разглашение ключа будет являться проблемой. Время жизни протокола разделено на периоды $1,\ldots, N$ (для простоты периодами времени являются отрезки равной длины, например, один день). К началу каждого периода пользователь взаимодействует с защищенным устройством для того, чтобы получить временный секретный ключ, который будет использован при расшифровании сообщений, отправленный в течение этого периода; пусть SK_i — временный ключ для периода i. С другой стороны, открытый ключ PK используется для зашифрования сообщений и является *неизменным* для всех периодов; вместо этого, шифртекстам ставится метка того

периода времени, в течение которого они были зашифрованы. Другими словами, результатом зашифрования сообщения M в период i является шифртекст $<i, C>$.

Незащищенное устройство, которое выполняет расшифрование данных, уязвимо к разглашению повторного ключа, т. е. через $t < N$ периодов может быть скомпрометирован ключ (где t — параметр). Задача схемы состоит в том, чтобы минимизировать влияние таких компрометаций. Когда ключ SK_i разглашен, злоумышленник сможет расшифровать все сообщения, отправленные в течение периода времени i. Предполагается, что злоумышленник сможет это сделать. Но при этом злоумышленник не сможет получить информацию обо всех остальных сообщениях, отправленных в любой другой период времени, чем в тот, в который произошла компрометация. Это и есть самый высокий уровень безопасности, который можно ожидать от такой модели. Схему, удовлетворяющую выше описанным условиям, называется схемой с изоляцией (t,N)-ключа.

Если физически безопасное устройство является полностью защищенным, то это устройство само генерирует (PK,SK^*), хранит ключ SK^* и публикует ключ PK. Когда пользователь запрашивает ключ для периода i, устройство может вычислить SK_i и отправить его пользователю. Более сложные методы нужны, когда физически безопасное устройство *не* является доверенным по отношению к пользователю. В этом случае используется более сложный алгоритм: пользователь может сам генерировать (PK,SK), публиковать ключ PK, и создавать ключи SK^* , SK_0. После этого пользователь отправляет SK^* на устройство и у себя хранит SK_0. Когда пользователь запрашивает ключ для периода i, устройство отправляет ему частичный ключ . Используя ключи SK_{i-1} и , пользователь вычисляет ключ периода SK_i. В этом способе безопасность пользователя гарантируется в течение *всех* периодов времени, касающихся самого устройства, поскольку знания ***только*** одного ключа SK^* недостаточно, чтобы получить любой из ключей периодов SK_i. Считается, что эта гарантия высокой безопасности является главной в том случае, когда единственное устройство обслуживает много различных пользователей, обеспечивая защиту их ключей против разглашения. В этом алгоритме пользователи могут доверять данному устройству при обновлении своих ключей, но при этом могут не желать, чтобы устройство имело способность читать их зашифрованный трафик. Поэтому нет причины, по которой устройство должно было бы иметь полное знание своих ключей периодов. Устройства синхронизируются в один и тот же период времени так, что только один секретный ключ за период выдается физически безопасным устройством. Они управляют аутентификационным взаимодействием, которое лежит в основе данного протокола.

Схема ЭЦП, стойкая к вторжениям [17], объединяет в себе лучшие черты трех подходов: опережающая безопасность, пороговая безопасность и безопасность с изоляцией ключа. А именно, при выполнении эффективных вычислений ЭЦП схема, стойкая к вторжениям, обеспечивает:

· безопасность прошлого и будущего периодов времени в случае, когда и подписывающая, и проверяющая стороны скомпрометированы не одновременно;

· безопасность в прошедший период времени, в случае одновременной компрометации и подписывающего, и проверяющего.

Функциональность различных компонентов системы, работающей по схеме ЭЦП, стойкой к вторжениям выглядит следующим образом: секретные ключи системы могут быть изменены двумя способами: обновлением (refresh) и изменением (update). Изменения секретных ключей происходит при переходе от одного периода времени к другому, изменяется также номер периода в ЭЦП. А обновления влияют только на внутренние сообщения.

Достоинства схемы ЭЦП, стойкой к вторжениям, в отличие от схемы с опережающей безопасностью и схемы с изоляцией ключа:

· независимое устройство (home base) обновляет свое состояние к концу каждого периода времени, а также отправляет обновленную информацию подписывающему;

· используется специальная процедура обновления: если после компрометации одного из модулей, но до компрометации другого, выполняется обновление, то информация, изученная противником в результате компрометации, становится бесполезной, следовательно, система остается защищенной (за исключением случая компрометации подписи в текущий период времени). Более того, поскольку обновление (refresh) связано только с одним сообщением от устройства к подписывающей стороне, это обновление можно объединить с полным обновлением (update) и затем запускать, по крайней мере, в любой период времени;

· если противник скомпрометирует только устройство (фактически, даже если устройство контролируется противником с самого начала), он все равно не сможет подделать подписи. А также, если противник скомпрометирует подписывающего, то он может подделать подписи, но только в те периоды времени, во время которых была получена секретная информация (либо в случае компрометации подписывающего, либо в случае комбинирования компрометации подписывающего и перехвата некоторых обновленных и измененных сообщений). Данная модель допускает многократные компрометации обоих сторон: и устройства, и подписывающего (в любом порядке) до тех пор, пока существует обновление между любой компрометацией различных модулей.

На основании обзора существующих решений в области дистанционного выполнения криптографических операций можно сделать заключение о том, что такие средства в настоящее время недостаточно развиты и имеют слабое распространение (слабо применяются на практике). Это вызывает необходимость дальнейших исследований в этой области и разработки комплексного подхода к дистанционному выполнению различных операций в рамках единого СКЗИ.

С учетом выбора предлагаемых методов и средств, обеспечивающих дистанционное выполнение криптографических операций и которые рассмотрены ранее, на данный момент можно выделить три протокола:

1) для шифрования данных — протокол RKEP;

2) для проверки целостности данных — протокол проверки целостности данных с использованием запросов к незащищенной памяти;

3) для генерации и проверки ЭЦП — протокол на основе схемы SiBIR.

Эти протоколы позволяют в совокупности организовать эффективную работу СКЗИ:

· протокол RKEP увеличивает скорость шифрования информации по сравнению с обычными методами;

· протокол проверки целостности данных с использованием запросов к незащищенной памяти значительно уменьшает количество запросов к памяти при проверке, не вызывая соответствующего увеличения используемого объема памяти;

· протокол на основе схемы SiBIR осуществляет многократное обновление ключевой информации на всех промежутках времени работы протокола.

Поскольку уязвимым местом в системе дистанционного выполнения криптографических операций является управление ключами, рассмотрим в общем виде ключевую систему СКЗИ.

Пусть $k_1, k_2, \ldots, k_n$ — «базовые» ключи, которые размещены на хорошо защищенных средствах, а k_d — «дистанционный» ключ, который размещен на плохо защищенном средстве, где ключ k_d образован из ключей $k_1, k_2, \ldots, k_n$ в результате применения однонаправленных функций. Следовательно, схему дистанционного управления ключами можно разделить на два вида:

1. «Статическая» — безопасность СКЗИ нарушается:

а) либо когда скомпрометированы все «базовые» ключи;

б) либо когда скомпрометирован «дистанционный» ключ.

2. «Динамическая» — безопасность СКЗИ нарушается:

а) либо когда скомпрометированы все «базовые» ключи только на интервале;

б) либо когда скомпрометирован «дистанционный» ключ только на интервале.

Очевидно, что недостатком в протоколах дистанционного шифрования и проверки целостности данных является использование «статической» схемы дистанционного управления ключами. Поэтому для повышения стойкости системы дистанционного выполнения криптографических операций протоколы усовершенствованы за счет изменения «статической» схемы на «динамическую».

Таким образом, предложенная СДВКО объединяет в себе несколько различных по своим свойствам и функциональному назначению протоколов. Для сведения в единый комплекс протоколов со статической и динамической генерацией ключа, а также в целях повышения стойкости разрабатываемой СДВКО к компрометации ключа в теоретическую основу разработки системы положена динамическая модель дистанционного генерации ключа.

СПИСОК ЛИТЕРАТУРЫ:

1. *Beller M. J., Chang L., Yacobi Y.* Privacy and Authentication in a Portable Communications System // IEEE Journal on Selected Areas in Communications. August 1993.
2. *Beaver D., Feigenbaum J., Shoup V.* Hiding Instances in Zero-Knowledge Proof Systems. Advances in Cryptology // Crypto '90, Lecture Notes in Computer Science. Springer, Berlin, 1991. Vol. 537. P. 326–338.
3. *Blaze M.* High-Bandwidth Encryption with Low-Bandwidth Smartcards. AT&T Bell Laboratories. 1995.
4. *Menezes A. J., van Oorschot P. C., Vanstone S. A.* Handbook of Applied Cryptography. CRC Press. 1996.
5. *Blum M., Evans W. S., Gemmell P., Kannan S., Naor M.* Checking the correctness of memories // Proc. Symp. Foundations of Comp. Science. October 1991. P. 90–99.
6. *Maheshwari U., Vingralek R., Shapiro W.* How to build a trusted database system on untrusted storage // Proc. USENIX Operating Sys. Design and Impl. Symp. October 2000. P. 135–150.
7. *Shapiro W., Vingralek R.* How to build a trusted database system on untrusted storage // Proc. Digital Rights Management Wkshp. January 2001. P. 176–191.
8. *Devanbu P. T., Stubbleline S.* G. Stack and queue integrity on hostile platforms // Software Engineering. January 2002. Vol. 28. P. 100–108.
9. *Chen B., Morris R.* Certifying program execution with secure processors // Proc. USENIX HotOS. Wkshp. May 2003.
10. *Gassend B., Suh G. E., Clarke D., van Dijk M., Devadas S.* Caches and merkle trees for efficient memory authentication // Proc. Intl. Symp. High Perf. Comp. Arch. February 2003. P. 295–306.
11. *Williams D., Sirer E.* G. Optimal parameter selection for efficient memory integrity verification using merkle hash trees // Proc. Intl. Symp. Network Comp. and Appl. July 2004. P. 383–388.
12. *Potlapally N. R.* Verifying Data Integrity with Few Queries to Untrusted Memory. Princeton University.
13. *Dodis Y., Katz J., Xu S., Yung M.* Key-Insulated Public-Key Cryptosystems. June 2002. http://eprint.iacr.org/2002/077.
14. *Варфоломеев А. А., Запечников С. В., Маркелов В. В., Пеленицын М. Б.* Интеллектуальные карты и криптографические особенности их применений в банковском деле. М., 2000. –188 с.
15. *Stallings W.* Cryptography and Network Security: Principles and Practice. Prentice Hall, 1998.
16. *Акритас А. Г.* Основы компьютерной алгебры с приложениями. М., 1994.
17. *Itkis G., Reyzin L.* SiBIR: Signer-Base Intrusion-Resilient Signatures, June 2002. http://eprint.iacr.org/2002/054.